MIPS 指令格式:

機器指令格式	組合語言格式	參與運算的元素
R-type	add sub and srl sll …	3 個暫存器 :rs rt rd
I-type	lw sw beq bne addi…	2 個暫存器和 16bits constant/address
J-type	J jal…	1 個 26bits address

R

Op (6bits)	Rs (5bits)	Rt (5bits)	Rd (5bits)	Shamt(5bits)	Funct(6bits)

I

Op(6bits)	Rs(5bits)	Rt(5bits)	Address/immediate(16bits)

J

Op(6bits)	26-bit address

Field	Length	Meaning
Op	6	Basic operation ,opcode
Rs	5	First register source operand
Rt	5	Second register source operand or register destination operand
Rd	5	Register destination oprand,get the result of the operation
Shamt	5	Shift amount
Funct	6	Selects the specific variant of the operation in the field
Addr./immd.	16	16-bit constant or address
Address	26	address

機器碼產生 Steps:

1. 指令屬於何種格式 R I J
2. 畫出對應的欄位 各個欄位有幾個 bits
3. 查表(opcode ,function,register)

Ex1:

Lw $t0,32($s2) [指令] → 機器碼

Op	Rs	Rt	16-bit number

35	18	8	32

$s2=18 放 rs 欄位 $t0=8 放 rt 欄位 位移量(offset)32 放於位址欄位
Rt 欄位位於 lw 指令中是指定目的暫存器欄位,存放載入結果

Ex2:
Jump 指令的 26-bits address 將目的位址 32 bits
最左邊 4bits 及最右邊 2bits 去除填入
=> 目的位址 除以 4 得 26-bit address 欄

Ex3:
Beq, bne
PC+4 (beq or bne 的下一個指令位址) +16-bit 欄 *4 = 目的地位址

Ex4:
sll ,srt:
rs 欄位 unused 因此設成 0
sll $t2,$s0,4 #reg $t2 = reg $s2<<4 bits

指令集設計原則
Simplicity favors requlority(簡單有致於一致性):
1. 所有指令保持一致大小(32bits)
2. 不同指令格式 暫存器欄位會在同一位置
3. 算數指令 一律 3 個運算元

Smaller is faster:
Mips 沒有很多暫存器 只有 32 個
Make common faster:
PC-相對定址於分支跳躍 ，立即定址法得到常數運算元
good design demands good compromises(好的設計好的折衷)
為了兼顧 較大記憶體位址和常數 ,和指令長度一致
⇨ **3 種指令格式**